UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/717,680	11/21/2000	Martyn S. Lovell	777.334US1	9114

41505 7590 11/01/2005

WOODCOCK WASHBURN LLP (MICROSOFT CORPORATION)
ONE LIBERTY PLACE - 46TH FLOOR
PHILADELPHIA, PA 19103

EXAMINER	
SHRADER, LAWRENCE J	
ART UNIT	**PAPER NUMBER**
2193	

DATE MAILED: 11/01/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/717,680	LOVELL ET AL.	
	Examiner	Art Unit	
	Lawrence Shrader	2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *09 September 2005*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-9, and 12 - 28* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-9, and 12 - 28* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to the RCE/amendment filed on 9/9/2005.

2. Claims 1 – 9, and 12 – 28 remain rejected, and claims 10 and 11 have been cancelled as requested by the Applicant. The arguments of the amendment of 9/9/2005 have been considered, but are moot in view of new applied art necessitated by the amendment made by the Applicant.

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 1, 2, 8 – 9; 15 – 20; and 22 – 27 are rejected under 35 U.S.C. 103(a) as being unpatentable over Coad et al., U.S. Patent 6,851,107 (hereinafter referred to as Coad) in view of Banning et al., U.S. Patent 5,485,567 (hereinafter referred to as Banning) and further in view of Rivlin, U.S. Patent 6,032,159.

In regard to claim 1, Coad discloses a visual programming environment:

"A source code editor...;"

A source code editor is disclosed (e.g., Figure 2, ref no. 208 and related text)

"A graphical design surface operable to display a graphical object representing actual code of the source code module;"

A graphical display window (design surface) of actual source code of the source code module is disclosed (e.g., Figure 13 and related text).

> *"wherein upon a change in the source code module, the change in the source code is immediately communicated to the graphical design surface and the graphical design surface is updated to reflect the change in the source code module, wherein the design surface displays the graphical object, the graphical object represents a database object, the design surface is operative to bind a particular database system to the database object, the database object further includes a database column, the source code module includes a variable, and the design surface is operative to bind the database column to the variable."*

Coad discloses a source code edit is reflected in the graphical window (column 2, lines 43 – 52), but does not disclose a graphical object representing a database object, including a database column bound to a variable. However, Banning discloses a graphical object representing a column of a database (Abstract; column 5, lines 19 – 62; Figure 2), but does not explicitly disclose binding a variable to a database column; however, Rivlin discloses binding a variable to a column in a database. Therefore, it would have been obvious to one skilled in the art at the time the invention was made to combine the editing and binding of objects in a graphical window as taught by Coad with the graphical object representing a database column as taught by Banning and bound to variables as is well known and taught by Rivlin, because one would be motivated to clearly and concisely convey particular aspects of a database to a user and make changes via a window of information as taught by Banning in the Abstract.

In regard to claim 2, incorporating the rejection of claim 1:

> *"...a change in the graphical design surface is immediately communicated to the source code editor..."*

Coad discloses modifying the object module results in an update to the source code (column 2, lines 43 – 52).

In regard to claim 8, incorporating the rejection of claim 1:

"...comprising at least one compiler..."

Coad discloses a compiler in column 5, lines 11 – 14.

In regard to claim 9, incorporating the rejection of claim 1:

"...the design surface is operative to bind the source code module to at last one compiler..."

Coad discloses a compile function that interacts with the graphical window in column 5, lines 11 – 14. It would be inherent that a software programming environment would necessarily bind the code to the compiler in order to obtain executable code.

In regard to claim 15, Coad discloses a visual programming environment:

"Creating a graphical object on a design surface, the graphical object representing actual code of a software module;"

Actual code of a software module is displayed as an object module (column 5, lines 11 – 14; e.g., Figure 13and associated text).

"Generating source code particular to the application type."

The source code is particular to the various projects and languages (column2, lines 53 – 61).

"Binding the graphical object to an application type;"

"wherein upon a change in the source code module, the change in the source code is immediately communicated to the graphical design surface and the graphical design surface is updated to reflect the change in the source code module, wherein the design surface displays the graphical object, the graphical object represents a database object, the design surface is operative to bind a particular database system to the database object, the database object further

includes a database column, the source code module includes a variable, and the design surface is operative to bind the database column to the variable."

Coad discloses a source code edit is reflected in the graphical window (column 2, lines 43 – 52) and discloses binding graphical objects to text-based programming language (e.g., Figure 13 and related text), but does not disclose a graphical object representing a database object, including a database column bound to a variable. However, Banning discloses a graphical object representing a column of a database (Abstract; column 5, lines 19 – 62; Figure 2), but does not explicitly disclose binding a variable to a database column; however, Rivlin discloses binding a variable to a column in a database. Therefore, it would have been obvious to one skilled in the art at the time the invention was made to combine the editing and binding of objects in a graphical window as taught by Coad with the graphical object representing a database column as taught by Banning and bound to variables as is well known and taught by Rivlin, because one would be motivated to clearly and concisely convey particular aspects of a database to a user and make changes via a window of information as taught by Banning in the Abstract.

In regard to claim 16, incorporating the rejection of claim 15:

"...wherein the application type is a source code compiler."

Coad discloses a compiler in column 5, lines 11 – 14.

In regard to claim 17, incorporating the rejection of claim 15:

"...wherein the application type is a database application."

Coad discloses a source code edit is reflected in the graphical window (column 2, lines 43 – 52), but does not disclose a graphical object representing a database

application. However, Banning discloses a graphical object representing a column of a database (Abstract; column 5, lines 19 – 62; Figure 2). Therefore, it would have been obvious to one skilled in the art at the time the invention was made to combine the editing and binding of objects in a graphical window as taught by Coad with the graphical object representing a database as taught by Banning, because one would be motivated to clearly and concisely convey particular aspects of a database to a user and make changes via a window of information as taught by Banning in the Abstract.

In regard to claim 18, incorporating the rejection of claim 15:

"...wherein the application type is a source code interpreter."

The Coad invention does not explicitly reference binding the object to an interpreter, but a compiler (column 5, lines 11 – 14). Therefore, it would have been obvious to one skilled in the art that an interpreter could have been specified to "compile" the code line-by-line in order to execute the code.

In regard to claim 19, incorporating the rejection of claim 15:

"Modifying the source code; and"

"Refreshing the design surface to update the graphical object to reflect the modification to the source code."

Source code editing is reflected in the graphical window (column 2, lines 43 – 52).

In regard to claim 20, incorporating the rejection of claim 15:

"Modifying the graphical object on the design surface; and"

"Refreshing the source code to reflect the modification to the graphical object."

Coad modifies the object module and the modification results in an update to the source code (column 2, lines 43 – 52).

In regard to claims 22 - 27 (the computer-readable medium), they are rejected for the corresponding reasons put forth in the rejection of claims 15 – 20 (the method).

5. Claims 3 – 7; 21; and 28 are rejected under 35 U.S.C. 103(a) as being unpatentable over Coad et al., U.S. Patent 6,851,107 (hereinafter referred to as Coad) in view of Banning et al., U.S. Patent 5,485,567 (hereinafter referred to as Banning) and further in view of Rivlin, U.S. Patent 6,032,159, and further in view of Washburn et al., U.S. Patent 5,157,779 (hereinafter referred to as Washburn).

In regard to claims 3, incorporating the rejection of claim 1:

"A change manager operative to manage versioning...;"

Coad discloses a visual programming environment using a text editor to enter and modify source code, but neither Coad nor Banning nor Rivlin teaches a version manager. However, Washburn discloses a compare module, which manages two versions of a file (column 2, lines 5 – 8; e.g., Figure 20). Therefore, it would have been obvious to one skilled in the art at the time the invention was made to modify visual programming environment in the disclosure of Coad combined with Banning and Rivlin, with the compare manager as taught by Washburn because the compare manager provides the capability to update, change or delete portions of the module

functionality in order to tailor the testing system to a particular computer system or computer software application as taught by Washburn in the Abstract.

"An application data store operative to store a previous version..."

Coad discloses a visual programming environment using a text editor to enter and modify source code, but neither Coad nor Banning nor Rivlin teaches a storage means for previous versions. However, Washburn discloses a data store module, which stores the master file (the previous version) of the file (column 2, lines 3 – 5; e.g., Figure 20). Therefore, it would have been obvious to one skilled in the art at the time the invention was made to modify visual programming environment in the disclosure of Coad combined with Banning and Rivlin with the data store module as taught by Washburn because the compare manager needs previous versions to work with, and an obvious means to store them would be deemed necessary by one skilled in the art.

In regard to claims 4 and 7, incorporating the rejection of claim 3:

"..the difference between the source code module and the previous version of the source code module is highlighted..."

Coad discloses a visual programming environment using a text editor to enter and modify source code, but neither Coad nor Banning nor Rivlin teaches highlighting a difference between the source code module and the previous version of the module. However, Washburn teaches highlighting textual differences in a text editor, as in claim 4 (column 12, lines 6 – 8), and highlighting differences graphically in the graphics window, as in claim 7 (column 10, lines 52 – 59; e.g., Figure 7d). Therefore, it would have been obvious to one skilled in the art at the time the invention was made to modify the source code editor and the graphical design surface in the teaching of Coad combined with

Banning and Rivlin with the text difference highlighting feature and the graphical difference highlighting feature as taught by Washburn because one skilled in the art would fine it advantageous to allow the user to efficiently determine at a glance the differences between two files, e.g., highlighting differences between files of different versions to give a quick visual difference.

In regard to claims 5 and 6, incorporating the rejection of claim 4.

"...the difference is highlighted using..."

Neither Coad nor Banning nor Rivlin teaches highlighting a difference between the source code module and the previous version in the code editor with either a squiggly line (claim5) or a tooltip bar (claim 6) to highlight the difference. Washburn does teach highlighting textual differences (column 12, lines 6 – 8), but does not specify the type of highlighting. However, official notice is taken that highlighting a segment of text is well known in the art and can be done in many ways, e.g., color change of text or background, font change, italicization, bolding, tooltip, squiggly underlines, straight underlines, etc. to name a few. Therefore, it would have been obvious to one skilled in the art at the time the invention was made to modify the source code editor and the graphical design surface in the teaching of Coad combined with Banning and Rivlin with the text difference highlighting feature and the graphical difference highlighting feature as taught by Washburn combined with the well known knowledge of various different types of highlighting, because one skilled in the art would fine it advantageous to allow the user to efficiently determine at a glance the differences between two files, e.g., highlighting differences between files of different versions to give a quick visual difference.

In regard to claim 21, incorporating the rejection of claim 15:

"...reading a template having a pre-configured software module from a datastore."

Coad discloses a visual programming environment using a text editor to enter and modify source code, but neither Coad nor Banning nor Rivlin teaches a storage means for previous versions. However, Washburn discloses a data store module, which stores the master file (the previous version) of the file (column 2, lines 3 – 5; e.g., Figure 20). Therefore, it would have been obvious to one skilled in the art at the time the invention was made to modify visual programming environment in the disclosure of Coad combined with Banning and Rivlin with the data store module as taught by Washburn because the compare manager needs previous versions to work with, and an obvious means to store them would be necessary.

In regard to claim 28 (the computer-readable medium), incorporating the rejection of claim 22, rejected for the corresponding reason given in the rejection of claim 21 (the method).

6. **Claim 12** is rejected under 35 U.S.C. 103(a) as being unpatentable over Coad et al., U.S. Patent 6,851,107 (hereinafter referred to as Coad) in view of Banning et al., U.S. Patent 5,485,567 (hereinafter referred to as Banning) and further in view of Rivlin, U.S. Patent 6,032,159, as applied to claim 1 above, and further in view of Peddada et al., U.S. Patent 6,031,533 (hereinafter referred to as Peddada) .

"...the binding is established through a drag-and-drop interface."

Neither Coad, Banning nor Rivlin teaches the use of a drag-and drop interface. However, Peddada teaches the well known technique of a drag-and-drop interface, used in this application to bind a graphics object to a program (column 12, lines 28 – 44). Therefore, it would have been obvious to one skilled in the art at the time the invention was made to implement the visual programming environment of Coad combined with Banning and Rivlinto display graphical objects from a database, modified by Peddada to provide a drag-and-drop feature to accomplish the added functions enabled by the above combinations, because the drag-and-drop feature further simplifies the programming function for users with little or no programming experience as suggested at column 12, lines 40 – 44 of Paddada.

7. **Claim 13** is rejected under 35 U.S.C. 103(a) as being unpatentable over Coad et al., U.S. Patent 6,851,107 (hereinafter referred to as Coad) in view of Banning et al., U.S. Patent 5,485,567 (hereinafter referred to as Banning) and further in view of Rivlin, U.S. Patent 6,032,159, as applied to claim 1 above, and further in view of Gupta et al., U.S. Patent 6,484,156 (hereinafter referred to as Gupta)

> *"...provide an interface to highlight a set of software modules that are grouped together as a package."*

Neither Coad, Banning nor Rivlin discloses highlighting a set of modules grouped together. However, Gupta discloses highlighting a set or sets of annotations (modules) for execution (column 14, lines 59 – 67). Therefore, it would have been obvious to one skilled in the art at the time the invention was made to enhance the visual programming environment taught by Coad in combination with Banning and Rivlin, with the feature of

highlighting a set of modules to be downloaded as taught by Gupta because the combination allows the user to select and run a set of modules via the graphics window without needing any detailed programming knowledge and offering convenient means of pre-selecting sets as taught at column 15, lines 1 - 54.

8. **Claim 14** is rejected under 35 U.S.C. 103(a) as being unpatentable over Coad et al., U.S. Patent 6,851,107 (hereinafter referred to as Coad) in view of Banning et al., U.S. Patent 5,485,567 (hereinafter referred to as Banning) and further in view of Rivlin, U.S. Patent 6,032,159, and further in view of Gupta et al., U.S. Patent 6,484,156 as applied to claim 13, and further in view of O'Donnell et al., U.S. Patent 6,223,203 (hereinafter referred to as O'Donnell)

> *"...receive a list of system identifiers...identifying a particular computer system..."*

Coad provides a visual programming interface, but neither Coad, Banning, Rivlin nor Gupta discloses a means to receive a list of system identifiers of a particular computer system to deploy a package. However, O'Donnell discloses a means to receive a list of system identifiers of a particular computer system (column 3, lines 1 – 20). Therefore, it would have been obvious to one skilled in the art at the time the invention was made to enhance the visual programming environment taught by Coad combined with Banning and Rivlin with the feature of highlighting a set of modules to be downloaded as taught by Gupta, and further modified with a received list of list of possible computer systems to select a particular system for module deployment as taught by O'Donnell, because this added feature obviously provides the user a means to select a

particular computer system to receive the software module set disclosed by the Gupta invention giving the user more detailed control over the system configuration.

Conclusion

9. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Lawrence Shrader whose telephone number is (571) 272-3734. The examiner can normally be reached on M-F 08:00-16:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (571) 272-3719. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Lawrence Shrader
Examiner
Art Unit 2193

October 25, 2005

KAKALI CHAKI
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100